UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/847,063	04/30/2001	Ming Zhou	GEI-001US 29083	4555

21718 7590 04/07/2005

LEE & HAYES PLLC
SUITE 500
421 W RIVERSIDE
SPOKANE, WA 99201

EXAMINER	
RUTTEN, JAMES D	

ART UNIT	PAPER NUMBER
2192	

DATE MAILED: 04/07/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/847,063	ZHOU ET AL.	
	Examiner	Art Unit	
	J. Derek Rutten	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *20 January 2005*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,7-11,13-15,18-24,26,32-35,39-43,49,51,52 and 55* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4,7-11,13-15,18-24,26,32-35,39-43,49,51,52 and 55* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *15 July 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Acknowledgement is made of Applicant's Request for Reconsideration dated 20 January 2005, responding to the 20 October Office action provided in the rejection of claims 1-4,7-11,13-15,18-24,26,32-35,39-43,49,51,52 and 55, wherein no claims have been amended, no claims have been canceled, and no new claims have been added. Claims 1-4,7-11,13-15,18-24,26,32-35,39-43,49,51,52 and 55 remain pending in the application and have been fully considered by the examiner.

2. Applicant's arguments, see page 5 of the Request for Reconsideration, filed 20 January 2005, with respect to the rejection of claims 1-4,7-11,13-15,18-24,26,32-35,39-43,49,51,52 and 55 under 35 U.S.C. § 103(a) have been fully considered and are persuasive in part. Therefore, the rejection has been withdrawn. However, upon further consideration, a new ground of rejection is made in view of "Gawk 3.1 new feature list" by Arnold D. Robbins, September 3, 2000 (art made of record).

3. Applicant's previous amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event,

however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Drawings

4. Applicant's arguments on pages 3 and 4 regarding the drawings have been considered and are persuasive. Therefore, the objection to the drawings has been withdrawn.

Response to Arguments

5. Applicant's arguments on page 5 line 22 to page 6 line 4 fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references. Furthermore, see art made of record "Gawk 3.1 new feature list" by Arnold D. Robbins, September 3 2000.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 1-4, 13-15, 18-24, 26, 32-35, 41-43, 49, 51, 52, and 55 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent 5,678,039 to Hinks et al. (hereinafter referred to as "Hinks"), in view of "OpenWindows Developer's Guide: Xview Code Generator

Programmer's Guide" by Sun Microsystems (hereinafter referred to as "Sun") further in view of "Gawk 3.1 new feature list" by Robbins (hereinafter "Robbins").

As per claim 1, Hinks discloses:

A computer-implemented method comprising:

analyzing a computer-servable document written for a particular locale (column 3 lines 7-9: "First, the sources are **parsed** by the Export/Import module to a translatable format."); *and*

extracting locale-sensitive content from the document while leaving locale-independent elements in the document (column 3 lines 1-4: "The Export/Import module itself includes a parsing engine to **extract** strings and translatable information from application programs."; also column 3 lines 39-43: "In either instance, the underlying program code (i.e., the code which the programmer has written to carry out the functionality of the program) has remained **untouched** by the process.").

storing the locale-sensitive content in a data structure separate from the document (column 3 lines 13-16).

Hinks does not expressly disclose the removal of locale-sensitive content or the substitution of a locale-sensitive content with a function call.

However, in an analogous environment, Sun teaches *removing locale-sensitive content* (page 100: "If you then use the GXV code generator to

generate the C source code, the interface's text strings get linked with the gettext() or dgettext() function call. These calls are used to look up the strings in other languages, or locales." Locale-sensitive content is removed through the process of linking the function calls in the code. The function calls use the text strings as keys to a text database of translations for other locales. As they are used as keys to a database and are no longer directly available to the original source code, the strings are no longer locale-sensitive.), *substituting, in a same operation as the extracting and removing, a function call in place of associated locale-sensitive content in the document* (page 100 as cited above: "...the interface's text strings get linked with the gettext() or dgettext() function call."), *the function call being configured such that, when executed, the function call obtains the locale-sensitive content from the data structure* (page 98 under the header "gettext() and dgettext() Routines").

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's function call substitution in Hinks' localization system. One of ordinary skill would have been motivated to include nothing specific to one's language and culture in software development and to provide features that facilitate translation of text into other languages.

Hinks and Sun do not expressly disclose substituting in a same operation as the extracting. However, in an analogous environment, Robbins teaches substituting in a same operation as extracting (page 1, item 3 in the list of changes: "3. New '--gen-

`po' option creates GNU gettext .po files for strings marked with a leading underscore.`"). It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Robbins' teaching of automatic extraction with Sun's function call insertion. One of ordinary skill would have been motivated to simplify program flow by combining multiple steps in one operation thereby easing the burden on the programmer.

As per claim 2, the above rejection of claim 1 is incorporated. Further, Hinks discloses *wherein the analyzing comprises examining each line of code in the document and based on the code, identifying the locale-sensitive content* (column 3 lines 1-4).

As per claim 3 the above rejection of claim 1 is incorporated. Further, Hinks discloses *wherein the locale-sensitive content comprises natural language text* (column 3 lines 28-31).

As per claim 4 the above rejection of claim 1 is incorporated. Further, Hinks discloses *wherein the locale-independent elements comprise source code and formatting data* (column 3 lines 31-34).

As per claim 13, Hinks discloses:

A method comprising:

automatically compiling a computer-servable document written for a particular locale to extract and remove any locale-sensitive content (column 3 lines 1-4, and 7-9 as cited in the above rejection of claim 1), *the compiling producing a compiled document with locale-independent elements* (FIG. 3 element 377 and column 8 lines 30-34: "Alternatively, a Resource Compiler 365, again such as Borland's Resource Workshop®, may be employed to **re-compile** the Translated Resource Files 360 and bind those compiled resources back into the target program, now shown as Translated Program 377."); *and*

storing the locale-sensitive content in a form that can be translated to other locales (column 3 lines 15-22: "From there, Export/Import (EXPIMP) module parses the resource file into a Translation Table, which is typically **stored** as a database table. The Translation Table encapsulates all the information that is known or can be derived from the various resources and stores them in a format which may be utilized by various **editors**."; also column 3 lines 53-55: "In this fashion, changing a product from one locale to another can be reduced to the simple process of swapping out resource files.").

Hinks does not expressly disclose the substitution of a locale-sensitive content with a function call.

However, in an analogous environment, Sun teaches *substituting a function call in place of the locale-sensitive content in the compiled document* (page 99 under the header "xgettext and msgfmt Utilities"), *the function call being configured such that, when executed, the function call obtains and embeds the locale-sensitive content back into the compiled document* (page 98 under the header "gettext() and dgettext() Routines").

All further limitations have been addressed in the above rejection of claim 1.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's function call substitution in Hinks' localization system. One of ordinary skill would have been motivated to include nothing specific to one's language and culture in software development and to provide features that facilitate translation of text into other languages.

As per claims 14 and 15, all limitations have been addressed in the above rejections of claims 3 and 4, respectively.

As per claim 18, the above rejection of claim 13 is incorporated. Further, Hinks discloses *retrieving, at runtime, the compiled document and populating the compiled document with the locale-sensitive content to reconstruct the computer-servable document that can be served to the particular locale* (column 3 lines 50-55).

As per claim 19, the above rejection of claim 13 is incorporated. Further, Hinks discloses *wherein the locale-sensitive content is translated in to a second version for use*

in a second locale (column 3 lines 35-36). All further limitations have been addressed in the above rejection of claim 18.

As per claim 20, Hinks discloses:

compiling a computer-servable document written for a particular locale to extract any locale-sensitive content, the compiling producing a compiled document with locale-independent elements (column 3 lines 1-4, and 7-9 as cited in the above rejection of claim 1; also FIG. 3 element 377 and column 8 lines 30-34 as cited in the above rejection of claim 13)*;*

storing the locale-sensitive content (column 3 lines 15-22 as cited in the above rejection of claim 13)*; and*

at runtime, retrieving the compiled document and populating the compiled document with the locale-sensitive content (column 3 lines 50-55 as cited in the above rejection of claim 18).

Further, Hinks discloses obtaining the associated locale-sensitive content and inserting the associated locale-sensitive content back into the compiled document (column 3 lines 50-55).

Hinks does not expressly disclose *substituting a function call in place of associated locale-sensitive content in the compiled document; and the populating comprises executing the function call in the compiled document.*

However, in an analogous environment, Sun teaches *substituting a function call in place of associated locale-sensitive content in the compiled document* (page 99 under the

header "xgettext and msgfmt Utilities" as cited in the above rejection of claim 6) *and the populating comprises executing the function call in the compiled document* (page 98 under the header "gettext() and dgettext() Routines" as cited in the above rejection of claim 6).

All further limitations have been addressed in the above rejection of claim 1.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's function call substitution and population in Hinks' localization system. One of ordinary skill would have been motivated to include nothing specific to one's language and culture in software development and to provide features that facilitate translation of text into other languages.

As per claims 21 and 22, the above rejection of claim 20 is incorporated. All further limitations have been addressed in the above rejection of claims 3 and 4, respectively.

As per claim 23, the above rejection of claim 20 is incorporated. Further, Hinks discloses *storing the locale-sensitive content in a structured text file* (column 3 lines 9-11. Resource files structured text files.).

As per claim 24, the above rejection of claim 20 is incorporated. Further, Hinks discloses *storing the locale sensitive content in a database file* (column 3 lines 13-16).

As per claim 26, the above rejection of claim 26 is incorporated. Hinks further discloses:

storing one or more translated versions of the locale-sensitive content (column 3 lines 23-24)*; and*

at runtime, retrieving the compiled document and populating the compiled document with a translated version of the locale-sensitive content (column 3 lines 50-55).

As per claim 32, Hinks discloses a system (FIG. 3). Hinks further discloses:

at least one computer-servable document stored in a computer-readable medium, the document being written for a particular locale (column 3 lines 7-9: "First, the **sources** are parsed by the Export/Import module to a translatable format."; also column 5 lines 59-61: "Software system 200, which is stored in system memory 102 and on **disk memory** 107, includes a kernel or operating system (OS) 240 and a windows shell 250."); *and*

a compiler to automatically extract locale-sensitive content from the document to produce a compiled document containing locale-independent elements (column 3 lines 7-7: "First, the sources are parsed by the **Export/Import module** to a translatable format."; also column 3 lines 50-51: "Resources for products to be translated are stored in an **external resource file** in a standard format.").

Hinks also discloses obtaining the associated locale-sensitive content and inserting the associated locale-sensitive content back into the compiled document (column 3 lines 50-55).

Hinks does not expressly disclose *substituting a function call in place of associated locale-sensitive content in the compiled document.* All further limitations have been addressed in the above rejection of claim 5.

However, in an analogous environment, Sun teaches *substituting a function call in place of associated locale-sensitive content in the compiled document* (page 99 under the header "xgettext and msgfmt Utilities" as cited in the above rejection of claim 6) *the function call being configured such that, when executed, the function call obtains the associated locale-sensitive content from the data structure and inserts the associated locale-sensitive content back into the compiled document* (page 98 under the header "gettext() and dgettext() Routines" as cited in the above rejection of claim 6).

All further limitations have been addressed in the above rejection of claim 1.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's function call substitution and population in Hinks' localization system. One of ordinary skill would have been motivated to include nothing specific to one's language and culture in software development and to provide features that facilitate translation of text into other languages.

As per claims 33 and 34, the above rejection of claim 32 is incorporated. All further limitations have been addressed in the above rejections of claims 3 and 4, respectively.

As per claim 35, the above rejection of claim 32 is incorporated. Hinks further discloses *wherein the compiler examines source code in the document to determine, from the source code, whether locale-sensitive content is present* (column 3 lines 7-9).

As per claim 41, Hinks discloses:

A compiler system (FIG. 3) *comprising:*

a grammar containing rules for structuring source code (column 3 lines 7-9 as cited in the above rejection of claim 1 describes a parser which inherently uses a source code grammar, otherwise it would be unable to match tokens of the source code.);

a call library to store function calls (column 5 lines 26-29 describe use of the Microsoft Windows environment, which inherently contains a call library);

content analyzer to analyze source code in a document written for a particular locale and to utilize the grammar to determine whether the source code contains locale-sensitive content that is specific to the particular locale (column 3 lines 7-9 as cited in the above rejection of claim 1),

Hinks does not expressly disclose *the content analyzer being configured to the locale-sensitive content in the source code.*

However, in an analogous environment, Sun teaches *the content analyzer being configured to the locale-sensitive content in the source code with associated references to the replaced locale-sensitive content* (page 99 under the header "xgettext and msgfmt Utilities", and page 98 under the header "gettext() and dgettext() Routines" as referenced in the above rejection of claim 6).

All further limitations have been addressed in the above rejection of claim 1.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's reference replacement in Hinks' localization system. One of ordinary skill would have been motivated to include nothing specific to one's language and culture in software development and to provide features that facilitate translation of text into other languages.

Hicks does not expressly disclose replacing locale-sensitive content with function calls. However, Sun teaches *the content analyzer replaces the locale-sensitive content with one or more function calls from the call library, the function calls being configured such that, when executed, the function calls reinsert the locale-sensitive content back into the source code* (page 99 under the header "xgettext and msgfmt Utilities" and page 98 under the header "gettext() and dgettext() Routines"). It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's function calls to reference Hinks' data structure. One of ordinary skill would have been motivated to retrieve text in the local language from a program with placeholders for a localizer to put each string's translation.).

As per claim 42, the above rejection of claim 41 is incorporated. Hinks further discloses *wherein the locale-sensitive content is placed in a separate file,* (column 3 lines 13-16).

Hinks does not expressly disclose *and the references comprise a pointer to that file.*

However, Sun teaches a reference to a file containing locale-sensitive content (page 97 under "Text Databases (Text Domains)".

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's reference pointer with Hinks' content file. One of ordinary skill would have been motivated to retrieve text in the local language from a program with placeholders for a localizer to put each string's translation.

As per claim 43, the above rejection of claim 41 is incorporated. Hinks further discloses *wherein the locale-sensitive content is placed in a separate data structure* (column 3 lines 13-16 as cited in claim 5).

Hinks does not expressly disclose references that comprise function calls.

However, Sun teaches *the references comprise function calls that, when executed, obtain the associated locale-sensitive content from the data structure and insert the associated locale-sensitive content into the source code* (page 99 under the header "xgettext and msgfmt Utilities" and page 98 under the header "gettext() and dgettext() Routines").

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's function calls to reference Hinks' data structure. One of ordinary skill would have been motivated to retrieve text in the local language from a program with placeholders for a localizer to put each string's translation.

As per claim 49, Hinks discloses:

A system (FIG. 3) comprising:

compilation means for compiling a computer-servable document written for a particular locale to extract any locale-sensitive content (column 3 lines 1-4, and 7-9 as cited in the above rejection of claim 1), *the compilation means producing a compiled document with locale-independent elements* (FIG. 3 element 377 and column 8 lines 30-34 as cited in the above rejection of claim 13)*; and*

storage means for storing the locale-sensitive content extracted from the computer-servable document in a data structure separate from the compiled document (column 3 lines 13-16 as cited in the above rejection of claim 5).

All further limitations have been addressed in the above rejection of claim 1.

Hinks does not expressly disclose substitution with a reference. However, Sun teaches all further limitations as have been addressed in the above rejection of claim 13. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's reference substitution in Hinks' localization system. One of ordinary skill would have been motivated to include nothing specific to one's language

and culture in software development and to provide features that facilitate translation of text into other languages.

As per claim 51, the above rejection of claim 49 is incorporated. All further limitations have been addressed in the above rejection of claim 18.

As per claim 52, the above rejection of claim 49 is incorporated. All further limitations have been addressed in the above rejection of claim 19.

As per claim 55, Hinks discloses:

One or more computer-readable media (column 5 line 44: "main memory") *comprising computer-executable instructions that, when executed, direct a computer to:*

examine source code in a document written for a particular locale (column 3 lines 7-9: "First, the sources are **parsed** by the Export/Import module to a translatable format.");

extract any locale-sensitive content from the source code (column 3 lines 1-4: "The Export/Import module itself includes a parsing engine to **extract** strings and translatable information from application programs.");

store the locale-sensitive content in a separate file (column 3 lines 13-16: "From there, Export/Import (EXPIMP) module parses the resource

file into a Translation Table, which is typically stored as a database table.").

Hinks does not expressly disclose: *substitute, in place of the locale-sensitive content in the document, function calls which when executed at runtime, re-supply the locale-sensitive content to the document.*

However, Sun teaches: *substitute, in place of the removed locale-sensitive content in the document, function calls which when executed at runtime, re-supply locale-sensitive content to the document*(page 99 under the header "xgettext and msgfmt Utilities" and page 98 under the header "gettext() and dgettext() Routines).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Sun's reference substitution in Hinks' localization system. One of ordinary skill would have been motivated to include nothing specific to one's language and culture in software development and to provide features that facilitate translation of text into other languages.

8. Claims 7, 8, and 10 are rejected under 35 U.S.C. 103(a) as being unpatentable over Sun in view of Robbins.

As per claim 7, Sun discloses:

A method comprising:

examining source code in a document written for a particular locale (page 97 "Text Databases": "To facilitate internationalization of text, a

process exists (see gettext(), later in this section) to **collect all text** visible to the user into a file called a portable object file. The portable object file contains the **native language** strings from a program and placeholders for a localizer to put each string's translation.");

determining, from the source code, locale-sensitive content that is specific to the particular locale (page 97 as cited above, "native language strings");

extracting and removing the locale-sensitive content from the source code (page 97 as cited above, "collect all text"; also page 100: "If you then use the GXV code generator to generate the C source code, the interface's text strings get linked with the gettext() or dgettext() function call. These calls are used to look up the strings in other languages, or locales." Locale-sensitive content is removed through the process of linking the function calls in the code. The function calls use the text strings as keys to a text database of translations for other locales. As they are used as keys to a database and are no longer directly available to the original source code, the strings are no longer locale-sensitive.);

storing the locale-sensitive content in a separate file (page 97 as cited above, "portable object file"); *and*

inserting, in a same operation as the removing, into the document in place of the removed locale-sensitive content, a reference to the locale-sensitive content in the separate file (page 98 under the header gettext() and dgettext() Routines: "Two

similar **routines** are available to a developer for **retrieving** translated text. One, gettext(), assumes a text domain has already been specified..."; page 99 under the header xgettext and msgfmt Utilities: "Once Devguide or a developer has **inserted** gettext() function calls around all user visible text in an application, xgettext can be run on the source files to produce the portable object files."), *wherein the reference comprises a function call that, when executed, obtains the locale-sensitive content from the separate file* (page 98 "retrieving translated text").

Sun does not expressly disclose substituting in a same operation as the extracting. However, in an analogous environment, Robbins teaches substituting in a same operation as extracting as pointed out in the above rejection of claim 1. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Robbins' teaching of automatic extraction with Sun's function call insertion. One of ordinary skill would have been motivated to simplify program flow by combining multiple steps in one operation thereby easing the burden on the programmer.

As per claim 8, the above rejection of claim 7 is incorporated. Sun further discloses *wherein the locale-sensitive content comprises natural language text* (page 97 "native language strings").

As per claim 10, the above rejection of claim 7 is incorporated. Sun further discloses *wherein the storing comprises storing the locale sensitive content in a structured text file* (page 99 "portable object file").

9. Claim 9 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sun and Robbins as applied to claim 7 above, and further in view of U.S. Patent Application Publication US 2002/0107684 to Gao, filed Feb. 7, 2001 (hereinafter referred to as "Gao").

As per claim 9, Sun does not expressly disclose ascertaining and identifying type.

However, in an analogous environment, Gao teaches internationalizing software using a semantic analysis phase:

ascertaining a type of code elements using a grammar for the source code (page 3, paragraph 51)*; and*

identifying, based on the type of the code elements, any locale-sensitive content delimited by the code elements (page 3, paragraph 52).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Gao's type analysis and identification determination in Sun's internationalization method. One of ordinary skill would have been motivated to detect potential internationalization problems using language grammar rules.

10. Claim 11 is rejected under 35 U.S.C. 103(a) as being unpatentable over Sun and Robbins as applied to claim 7 above, and further in view of Hinks.

As per claim 11, Sun does not expressly disclose storing content in a database file.

However, Hinks teaches the use of a database file for storage of locale sensitive content (column 3 lines 13-16).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to store Sun's localization data in Hinks database file. One of ordinary skill would have been motivated to store localization data in an easily searchable data structure.

11. Claims 39 and 40 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hinks, Sun, and Robbins as applied to claim 32 above, and further in view of Gao.

As per claim 39, the above rejection of claim 32 is incorporated. Further, Hinks does not expressly disclose a runtime manager.

However, Gao teaches the use of an "Integrated Translation Environment" which populates web documents with locale-sensitive content prior to serving (page 5 paragraph 0135).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Gao's web localizer with Hinks' internationalization method. One of ordinary skill would have been motivated to present localized content in a document.

As per claim 40, the above rejection of claim 32 is incorporated. Hinks further discloses a translated version of the locale-sensitive content (column 3 lines 23-24).

Hinks does not expressly disclose a runtime manager.

However, Gao teaches the use of an "Integrated Translation Environment" which populates web documents with locale-sensitive content during runtime (page 5 paragraph 0135).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Gao's web localizer with Hinks' internationalization method. One of ordinary skill would have been motivated to present localized content in a document when it is presented.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to J. Derek Rutten whose telephone number is (571) 272-3703. The examiner can normally be reached on T-F 6:00 - 4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

jdr

TUAN DAM
SUPERVISORY PATENT EXAMINER